```
call void @llvm.dbg.value(metadata ptr %a, metadata !21, metadata
                                                                                                                                                                                              ...!DIExpression()),!dbg!28
                                                                                                                                                                                              call void @llvm.dbg.value(metadata ptr %b, metadata !22, metadata
                                                                                                                                                                                              ...!DIExpression()),!dbg!28
                                                                                                                                                                                              call void @llvm.dbg.value(metadata ptr %c, metadata !23, metadata
                                                                                                                                                                                              ...!DIExpression()),!dbg!28
                                                                                                                                                                                              call void @llvm.dbg.value(metadata ptr %cond, metadata !24, metadata
                                                                                                                                                                                              ...!DIExpression()),!dbg!28
                                                                                                                                                                                              call void @llvm.dbg.value(metadata i32 %n, metadata !25, metadata
                                                                                                                                                                                              ...!DIExpression()),!dbg!28
                                                                                                                                                                                              tail call void asm sideeffect ".inst 0x2520e020", ""() #7, !dbg !29, !srcloc
                                                                                                                                                                                              call void @llvm.dbg.value(metadata i32 0, metadata !26, metadata
                                                                                                                                                                                              ...!DIExpression()),!dbg!32
                                                                                                                                                                                              %cmp12 = icmp sgt i32 %n, 0, !dbg !33
                                                                                                                                                                                              br i1 %cmp12, label %for.body.preheader, label %for.cond.cleanup, !dbg !35
                                                                                                                                                      for.body.preheader:
                                                                                                                                                      %wide.trip.count = zext i32 %n to i64, !dbg !33
                                                                                                                                                      %0 = call i64 @llvm.vscale.i64(), !dbg !35
                                                                                                                                                      %1 = \text{mul } i64 \%0, 2, !dbg !35
                                                                                                                                                      %2 = icmp uge i64 %wide.trip.count, %1, !dbg !35
br i1 %2, label %pre.alc, label %Preheader.for.remaining.iterations, !dbg !35
                                                                                                    pre.alc:
                                                                                                    \%7 = \text{call} < \text{vscale x 2 x i1} > \text{@llvm.aarch64.sve.ptrue.nxv2i1(i32 2)}
                                                                                                     %uniform.vector = call <vscale x 2 x i64>
                                                                                                    ... @llvm.experimental.stepvector.nxv2i64()
%8 = urem i64 %wide.trip.count, %1
                                                                                                     %total.iterations.to.be.vectorized = sub i64 %wide.trip.count, %8
                                                                                                     \%9 = insertelement <vscale x 2 x i64> poison, i64 \%1, i64 0
                                                                                                     %stepVector.update.values = shufflevector < vscale x 2 x i64 > %9, < vscale x 2
                                                                                                     ... x i64> poison, <vscale x 2 x i32> zeroinitializer
                                                                                                    %10 = getelementptr inbounds i32, ptr %cond, i64 0, !dbg !40
%11 = load <vscale x 2 x i32>, ptr %10, align 8
%12 = icmp eq <vscale x 2 x i32> %11, zeroinitializer
%13 = call i64 @llvm.aarch64.sve.cntp.nxv2i1(<vscale x 2 x i1> %12, <vscale
                                                                                                    ... x 2 x i1> %12)
                                                                                                     br label %alc.header
                                                                                      alc.header:
                                                                                       %index = phi i64 [ 0, %pre.alc ], [ %52, %new.latch ] %uniform.vector1 = phi <vscale x 2 x i64> [ %uniform.vector, %pre.alc ], [
                                                                                       ... %49, %new.latch ]
                                                                                       %uniform.vector.predicates = phi <vscale x 2 x i1> [ %12, %pre.alc ], [ %50,
                                                                                       ... %new.latch ]
                                                                                       %uniform.vec.actives = phi i64 [ %13, %pre.alc ], [ %51, %new.latch ]
                                                                                       %14 = call <vscale x 2 x i64> @llvm.aarch64.sve.index.nxv2i64(i64 %index,
                                                                                       ... i64 %1)
                                                                                       %15 = getelementptr inbounds i32, ptr %cond, i64 %index, !dbg !40
                                                                                       %16 = load <vscale x 2 x i32>, ptr %15, align 8
                                                                                       %17 = icmp eq < vscale x 2 x i32 > %16, zeroinitializer
                                                                                       %18 = call i64 @llvm.aarch64.sve.cntp.nxv2i1(<vscale x 2 x i1> %17, <vscale
                                                                                       ... x 2 x i1 > \%17
                                                                                       %19 = add i64 %uniform.vec.actives, %18
                                                                                       %condition2 = icmp ule i64 %19, %1
                                                                                       br i1 %condition2, label %lane.gather, label %linearized
                           lane.gather:
                           %20 = call <vscale x 2 x i64> @llvm.aarch64.sve.compact.nxv2i64(<vscale x 2
                           ... x i1> %uniform.vector.predicates, <vscale x 2 x i64> %uniform.vector1)
                            %21 = call <vscale x 2 x i64> @llvm.aarch64.sve.compact.nxv2i64(<vscale x 2
                            ... x i1> %17, <vscale x 2 x i64> %14)
                           %22 = call <vscale x 2 x i1> @llvm.aarch64.sve.whilelt.nxv2i1.i64(i64 0, i64
                            ... %uniform.vec.actives)
                           %23 = call <vscale x 2 x i64> @llvm.aarch64.sve.splice.nxv2i64(<vscale x 2 x ... i1> %22, <vscale x 2 x i64> %20, <vscale x 2 x i64> %21)
                            %24 = call <vscale x 2 x i1> @llvm.aarch64.sve.whilelt.nxv2i1.i64(i64 0, i64
                            ... %19)
                           %25 = icmp uge i64 %19, %1
br i1 %25, label %uniform.block, label %join.block
                                                                                                   F
uniform.block:
%26 = getelementptr inbounds i32, ptr %a, i64 0, !dbg !48
%27 = getelementptr inbounds i32, ptr %b, i64 0, !dbg !50
%28 = getelementptr inbounds i32, ptr %c, i64 0, !dbg !52
 %29 = \text{call} < \text{vscale } x \ 2 \ x \ i32 > 
... @llvm.aarch64.sve.ld1.gather.index.nxv2i32(<vscale x 2 x i1> %7, ptr %26, ... <vscale x 2 x i64> %23)
                                                                                                                                       linearized:
                                                                                                                                       %38 = getelementptr inbounds i32, ptr %a, i64 %index, !dbg !48
%30 = \text{call} < \text{vscale x 2 x i32} >
... @llvm.aarch64.sve.ld1.gather.index.nxv2i32(<vscale x 2 x i1> %7, ptr %27,
                                                                                                                                       %39 = getelementptr inbounds i32, ptr %b, i64 %index, !dbg !50
                                                                                                                                       %40 = getelementptr inbounds i32, ptr %c, i64 %index, !dbg !52 %41 = load <vscale x 2 x i32>, ptr %38, align 8
... <vscale x 2 x i64> %23)
%31 = \text{mul} < \text{vscale x 2 x i32} > %30, %29
                                                                                                                                       %42 = load <vscale x 2 x i32>, ptr %39, align 8
%43 = mul <vscale x 2 x i32> %42, %41
call void @llvm.aarch64.sve.st1.scatter.index.nxv2i32(<vscale x 2 x i32>
... %31, <vscale x 2 x i1> %7, ptr %28, <vscale x 2 x i64> %23)
                                                                                                                                       store <vscale x 2 x i32> %43, ptr %40, align 8
%32 = add i64 %index, %1
%33 = call <vscale x 2 x i64> @llvm.aarch64.sve.index.nxv2i64(i64 %32, i64 1) %34 = getelementptr inbounds i32, ptr %cond, i64 %32, !dbg !40 %35 = load <vscale x 2 x i32>, ptr %34, align 8
                                                                                                                                        br label %new.latch
%36 = icmp eq < vscale x 2 x i32 > %35, zeroinitializer
%37 = call i64 @llvm.aarch64.sve.cntp.nxv2i1(<vscale x 2 x i1> %36, <vscale
... x 2 x i1> %36)
br label %join.block
                            join.block:
                            \%44 = \text{phi i} 64 \text{ [ \%index, \%lane.gather ], [ \%32, \%uniform.block ]}
                            \%45 = phi < vscale \times 2 \times i64 > [\%uniform.vector1, \%lane.gather], [\%33,
                            ... %uniform.block ]
                            %46 = phi <vscale x 2 x i1> [ %uniform.vector.predicates, %lane.gather ], [
                             ... %36, %uniform.block ]
                            %47 = phi i64 [ %uniform.vec.actives, %lane.gather ], [ %37, %uniform.block ]
                            br label %new.latch
                                                                                                            new.latch:
                                                                                                             %48 = phi i64 [ %44, %join.block ], [ %index, %linearized ]
                                                                                                             %49 = phi < vscale \times 2 \times i64 > [\%45, \%join.block], [\%uniform.vector1,]
                                                                                                             .. %linearized 1
                                                                                                             %50 = phi < vscale x 2 x i1 > [ %46, %join.block ], [
                                                                                                            ... %uniform.vector.predicates, %linearized ]
%51 = phi i64 [ %47, %join.block ], [ %uniform.vec.actives, %linearized ]
%52 = add i64 %48, %1
                                                                                                            %53 = icmp ugt i64 %52, %8
br i1 %53, label %alc.header, label %middel.block
                                                                                                                                             middel.block:
                                                                                                                                             %condition = icmp eq i64 %8, 0
br i1 %condition, label %for.cond.cleanup.loopexit, label
                                                                                                                                             ... %Preheader.for.remaining.iterations
                                                                                                                                                                    Preheader.for.remaining.iterations:
                                                                                                                                                                     %6 = phi i64 [ 0, %for.body.preheader ], [ %52, %middel.block ]
                                                                                                                                                                     br label %for.body
                                                                                                                                                           for.body:
                                                                                                                                                            %indvars.iv = phi i64 [ %indvars.iv.next, %for.inc ], [ %6,
                                                                                                                                                            ... %Preheader.for.remaining.iterations ]
                                                                                                                                                            call void @llvm.dbg.value(metadata i64 %indvars.iv, metadata !26, metadata
                                                                                                                                                            ... !DIExpression()), !dbg !32
                                                                                                                                                           %arrayidx = getelementptr inbounds i32, ptr %cond, i64 %indvars.iv, !dbg !40 %3 = load i32, ptr %arrayidx, align 4, !dbg !40, !tbaa !43 %tobool.not = icmp eq i32 %3, 0, !dbg !40 br i1 %tobool.not, label %for.inc, label %if.then, !dbg !47
                                                                                                                                                                 if.then:
                                                                                                                                                                  %arrayidx2 = getelementptr inbounds i32, ptr %a, i64 %indvars.iv, !dbg !48
                                                                                                                                                                 %4 = load i32, ptr %arrayidx2, align 4, !dbg !48, !tbaa !43
%arrayidx4 = getelementptr inbounds i32, ptr %b, i64 %indvars.iv, !dbg !50
%5 = load i32, ptr %arrayidx4, align 4, !dbg !50, !tbaa !43
%mul = mul nsw i32 %5, %4, !dbg !51
                                                                                                                                                                  %arrayidx6 = getelementptr inbounds i32, ptr %c, i64 %indvars.iv, !dbg !52 store i32 %mul, ptr %arrayidx6, align 4, !dbg !53, !tbaa !43 br label %for.inc, !dbg !54
                                                                                                                                                                  for.inc:
                                                                                                                                                                   %indvars.iv.next = add nuw nsw i64 %indvars.iv, 1, !dbg !55
                                                                                                                                                                   call void @llvm.dbg.value(metadata i64 %indvars.iv.next, metadata !26,
                                                                                                                                                                   ... metadata !DIExpression()), !dbg !32
%exitcond.not = icmp eq i64 %indvars.iv.next, %wide.trip.count, !dbg !33
                                                                                                                                                                   br i1 %exitcond.not, label %for.cond.cleanup.loopexit, label %for.body, !dbg
                                                                                                                                                                   ... !35, !llvm.loop !56
                                                                                                                                                                                                                                       F
                                                                                                                                                                  for.cond.cleanup.loopexit:
                                                                                                                                                                  br label %for.cond.cleanup, !dbg !36
                                                                                                                                                                                     for.cond.cleanup:
                                                                                                                                                                                      tail call void asm sideeffect ".inst 0x2520e040", ""() #7, !dbg !36, !srcloc
                                                                                                                                                                                      ... !38
```

ret void, !dbg !39